Amendments to the Claims:

1. (Currently amended) A computer-implemented method for software testing, comprising:

analyzing the source code under test to generate ~~a scan file, including~~ a map of the source code under test [[and]] ~~or~~ a tree structure for the source code under test, and to generate a scan file of all or a portion of the map or tree structure of the source code under test;

generating a stub file for ~~the source code under test~~ all or a portion of the map or tree structure based on the ~~scan file~~ map or tree structure of the source code under test;

generating a driver test script file for all or a portion of the map or tree structure of the source code under test based on the scan file and the stub file;

generating a test driver for all or a portion of the map or tree structure of the source code under test based on the driver test script file; and

running the test driver on all or a portion of the map or tree structure of the source code under test to generate a results file for summarizing the results of the software testing.

2. (Original) The method of claim 1, wherein said analyzing step is automatically performed by a code scanner configurable by a graphical user interface.

3. (Original) The method of claim 1, wherein the step of generating the stub file is automatically performed by a stub generator configurable by a graphical user interface.

4. (Original) The method of claim 1, wherein the step of generating the driver test

script file is automatically performed by a driver script generator configurable by a graphical user interface.

5. (Original) The method of claim 1, wherein the step of generating the test driver is automatically performed in a driver build directory configurable by a graphical user interface.

6. (Original) The method of claim 1, further comprising displaying the results file with a graphical user interface.

7. (Original) The method of claim 1, wherein the map includes a list of executable source code names, package names, and procedure or function names for the package names.

8. (Original) The method of claim 7, further comprising generating the list of executable source code names for the stub file and the source code under test.

9. (Original) The method of claim 7, further comprising generating the tree structure by performing calls until the procedures or function for each of the packages are determined.

10. (Original) The method of claim 1, further comprising providing a graphical user interface for performing the analyzing step, the step of generating the stub file, the step of generating the driver test script file, the step of generating the test driver, and the running step.

11. (Original) The method of claim 10, further comprising employing the graphical

user interface for specifying variable and variable type definitions.

12. (Original) The method of claim 10, further comprising employing the graphical user interface for documenting the software testing.

13. (Original) The method of claim 10, further comprising providing a FTP capability via the graphical user interface.

14. (Original) The method of claim 10, further comprising providing multi-language support for the source code under test via the graphical user interface.

15. (Original) The method of claim 10, further comprising providing mixed-language software testing via the graphical user interface.

16. (Original) The method of claim 1, further comprising implementing the method via object-oriented programming.

17. (Original) The method of claim 10, further comprising displaying the scan file, the source code under test, the driver test script file, the test driver, and the results file, via the graphical user interface.

18. (Currently amended) A system for software testing, comprising:

means for analyzing the source code under test to generate a scan file, including a map of the source code under test [[and]] or a tree structure for the source code under test, and to

generate a scan file of all or a portion of the map or tree structure of the source code under test;

means for generating a stub file for ~~the source code under test~~ all or a portion of the map or tree structure based on the ~~scan file~~ map or tree structure of the source code under test;

means for generating a driver test script file for all or a portion of the map or tree structure of the source code under test based on the scan file and the stub file;

means for generating a test driver for all or a portion of the map or tree structure of the source code under test based on the driver test script file; and

means for running the test driver on all or a portion of the map or tree structure of the source code under test to generate a results file for summarizing the results of the software testing.

19. (Original) The system of claim 1, wherein said analyzing means is automatically activated by a graphical user interface.

20. (Original) The system of claim 18, wherein the means for generating the stub file is automatically activated by a graphical user interface.

21. (Original) The system of claim 18, wherein the means for generating the driver test script file is automatically activated by a graphical user interface.

22. (Original) The system of claim 18, wherein the means for generating the test driver is automatically activated by a graphical user interface.

23. (Original) The system of claim 18, further comprising means for displaying the results file.

24. (Original) The system of claim 18, wherein the map includes a list of executable source code names, package names, and procedure or function names for the package names.

25. (Original) The system of claim 24, further comprising means for generating the list of executable source code names for the stub file and the source code under test.

26. (Original) The system of claim 24, further comprising means for generating the tree structure by performing calls until the procedures or function for each of the packages are determined.

27. (Original) The system of claim 18, further comprising means providing a user interface for accessing the analyzing means, the means for generating the stub file, the means for generating the driver test script file, and the means for generating the test driver, and the running means.

28. (Original) The system of claim 27, further comprising means for employing the user interface for specifying variable and variable type definitions.

29. (Original) The system of claim 27, further comprising means for employing the user interface for documenting the software testing.

30. (Original) The system of claim 27, further comprising means for providing a FTP capability via the user interface.

31. (Original) The system of claim 27, further comprising means for providing multi-language support for the source code under test via the user interface.

32. (Original) The system of claim 27, further comprising means for providing mixed-language software testing via the user interface.

33. (Original) The system of claim 18, further comprising means for implementing the system via object-oriented programming.

34. (Original) The system of claim 27, further comprising means for displaying the scan file, the source code under test, the driver test script file, the test driver, and the results file, via the user interface.

35. (Original) The system of claim 27, wherein the user interface includes a graphical user interface.

36. (Original) The system of claim 18, wherein the means for analyzing, the means for generating the stub file, the means for generating the driver test script file, the means for generating the test driver, and the means for running the test driver comprise devices of a computer system.

37. (Currently amended) The system of claim 18, wherein the means for analyzing, the means for generating the stub file, the means for generating the driver test script file, the means for generating the test driver, and the means for running the test driver comprise computer-readable instructions stored on a tangible computer readable medium.